A Bayesian Hierarchical Model for Comparing Average F1 Scores

Dell Zhang
DCSIS
Birkbeck, University of London
Malet Street
London WC1E 7HX, UK
Email: dell.z@ieee.org

Jun Wang, Xiaoxue Zhao
Dept of Computing Science
University College London
Gower Street
London WC1E 6BT, UK
Email: {j.wang,x.zhao}@cs.ucl.ac.uk

Xiaoling Wang
Software Engineering Institute
East China Normal University
3663 North Zhongshan Road
Shanghai 200062, China
Email: xlwang@sei.ecnu.edu.cn

***Abstract*—In multi-class text classification, the performance (effectiveness) of a classifier is usually measured by micro-averaged and macro-averaged F1 scores. However, the scores themselves do not tell us how reliable they are in terms of forecasting the classifier's future performance on unseen data. In this paper, we propose a novel approach to explicitly modelling the uncertainty of average F1 scores through Bayesian reasoning, and demonstrate that it can provide much more comprehensive performance comparison between text classifiers than the traditional frequentist null hypothesis significance testing (NHST).**

***Keywords*-text classification; performance evaluation; hypothesis testing; model comparison; Bayesian inference**

I. Introduction

Automatic text classification [1] is a fundamental technique in information retrieval (IR) [2]. It has many important applications, including topic categorisation, spam filtering, sentiment analysis, message routing, language identification, genre detection, authorship attribution, and so on. In fact, most modern IR systems for search, recommendation, or advertising contain multiple components that use some form of text classification.

The most widely used performance measure for text classification is the F_1 score [3] which is defined as the harmonic mean of precision and recall. It is known to be more informative and more useful than classification accuracy etc. due to the prevalent phenomenon of class imbalance in text classification. When multiple classes exist in the document collection (such as Reuters-21578 with its 118 classes), we often want to compute a single aggregate measure that combines the F_1 scores for individual classes. There are two methods to do this: micro-averaging and macro-averaging [2]. The former pools per-document decisions across classes, and then computes the overall F_1 score on the pooled contingency table. The latter just computes a simple average of the F_1 scores over classes. The differences between these two averaging methods can be large: micro-averaging gives equal weight to each per-document classification decision and therefore is dominated by large classes, whereas macro-averaging gives equal weight to each class. It is nowadays a common practice for IR researchers to evaluate a multi-class text classifier using both the micro-averaged F_1 score (denoted as miF_1) and the macro-averaged F_1 score (denoted as maF_1), since their introduction by Yang and Liu's seminal SIGIR-1999 paper [4].

However, the average F_1 scores themselves only reflect a text classifier's performance on the given test data. How can we be sure that it will work well on unseen data? Given any finite amount of test results, we can never be guaranteed that one classifier's performance will definitely achieve a certain acceptable level (say 0.80) in practice. For example, suppose that a classifier got miF_1 0.81 on 100 test documents. Due to the small number of test documents, we probably do not have much confidence in pronouncing that its future performance will definitely be above 0.80. If instead the classifier got miF_1 0.81 on 100,000 test documents, we can be more confident than in the previous case. Nevertheless, there will always be some degree of uncertainty. The central question here is how to assess the uncertainty of a classifier's performance as measured by miF_1 and maF_1. Perhaps the simplest solution is to apply k-fold cross-validation [5] and then calculate the sample variance of average F_1 scores over multiple "folds" of the dataset. This method tends to yield poor estimations though: the sample variance can approximate the true variance well only if we have a large number of folds, but when the dataset is divided into many folds, the size of each fold is likely to be too small to give a meaningful average F_1 score. Hence it is desirable to derive the uncertainty of average F_1 scores directly from all the atomic document-category classification results.

In this paper, we build up on our previous preliminary work [6] to address this problem through Bayesian hierarchical modelling [7], [8], and demonstrate that our proposed Bayesian approach can provide much more comprehensive performance comparison between text classifiers than the traditional frequentist null hypothesis significance testing (NHST).

The rest of this paper is organised as follows. In Section II, we review the existing approaches to the problem of classifier performance comparison. In Section III, we present our Bayesian estimation based approach in detail. In Section IV, we show how the proposed approach can be used

to compare some well-known text classification algorithms. In Section V, we discuss several natural extensions to the proposed approach. In Section VI, we draw conclusions.

II. Related Work

A. Frequentist Performance Comparison

The traditional frequentist approach to comparing classifiers is to use NHST [5]. The usual process of NHST consists of four steps: (1) formulate the null hypothesis H_0 that the observations are the result of pure chance and the alternative hypothesis H_1 that the observations show a real effect combined with a component of chance variation; (2) identify a test statistic that can be used to assess the truth of H_0; (3) compute the p-value, which is the probability that a test statistic equal to or more extreme than the one observed would be obtained under the assumption of hypothesis H_0; (4) if the p-value is less than an acceptable significance level, the observed effect is statistically significant, i.e., H_0 is ruled out and H_1 is valid.

Specifically for performance comparison of text classifiers, the usage of NHST has been presented in detail by Yang and Liu in their SIGIR-1999 paper [4]. In summary, on the document level (micro level), sign-test can be used to compare two classifiers' accuracy scores (called s-test), while unpaired t-test can be used to compare two classifiers' performance measures in the form of proportions, e.g., precision, recall, error, and accuracy (called p-test); on the category level (macro level), sign-test and paired-t test can both be used to compare two classifiers' F_1 scores (which are called S-test and T-test respectively).

In spite of being useful and influential, such a frequentist approach unfortunately has many inherent deficiencies and limitations [8], [9]. To name a few: (i) NHST is only able to tell us whether the experimental data are sufficient to reject the null hypothesis (that the performance difference is zero) or not, but there is no way to accept the null hypothesis, i.e., it is impossible for us to confidently claim that two classifiers perform equally well; (ii) NHST will reject the null hypothesis as long as the experimental data suggest that the performance difference is non-zero, even if the performance difference is too slight to have any real effect in practice; (iii) complex performance measures such as the F_1 score can only be compared on the category level but not on the document level, which seriously restricts the statistical power of NHST as the number of categories is usually much much smaller than the number of documents.

The other NHST methods that have been applied to compare classifiers include ANOVA test [10], Friedman test [11], McNemar's test [12], and Wilcoxon signed-rank test [13]. Due to their frequentist nature, no matter which specific test they use, more or less they suffer from the above mentioned perils.

B. Bayesian Performance Comparison

1) Bayes Factor: Model comparison using Bayes factor has been applied to the problem of classifier performance comparison [14], [15]. In our context, the Bayes factor is the marginal likelihood of classification results data for the null model $\Pr[\mathcal{D}|\mathcal{M}_0]$ (where two classifiers perform equally well) relative to the marginal likelihood of classification results data for the alternative model $\Pr[\mathcal{D}|\mathcal{M}_1]$ (where one classifier works better than the other classifier): $\text{BF} = \Pr[\mathcal{D}|\mathcal{M}_0]/\Pr[\mathcal{D}|\mathcal{M}_1]$.

As the BF becomes larger, the evidence increases in favour of model $\mathcal{M}_0$ over model $\mathcal{M}_1$. The rule of thumb for interpreting the magnitude of the BF is that there is "substantial" evidence for the null model $\mathcal{M}_0$ when the BF exceeds 3, and similarly, "substantial" evidence for the alternative model $\mathcal{M}_1$ when the BF is less than $\frac{1}{3}$.

Although for simple models the value of Bayes factor can be derived analytically as shown by [14]–[17], for complex models it can only be computed numerically using for example the Savage-Dickey (SD) method [18]–[20]. The SD method assumes that the prior on the variance in the null model equals the prior on the variance in the alternative model at the null value: $\Pr[\sigma^2|\mathcal{M}_0] = \Pr[\sigma^2|\mathcal{M}_1, \delta = 0]$. From this it follows that the likelihood of the data in the null model equals the likelihood of the data in the alternative model at the null value: $\Pr[\mathcal{D}|\mathcal{M}_0] = \Pr[\mathcal{D}|\mathcal{M}_1, \delta = 0]$. Thus, the Bayes factor can be determined by considering the posterior and prior of the alternative hypothesis alone, because the Bayes factor is just the ratio of the probability density at $\delta = 0$ in the posterior relative to the probability density at $\delta = 0$ in the prior: $\text{BF} = \Pr[\delta = 0|\mathcal{M}_1, \mathcal{D}]/\Pr[\delta = 0|\mathcal{M}_1]$. The posterior density $\Pr[\delta = 0|\mathcal{M}_1, \mathcal{D}]$ and the prior density $\Pr[\delta = 0|\mathcal{M}_1]$ can both be approximated by fitting a smooth function to the Markov Chain Monte Carlo (MCMC) [7], [8] samples via kernel density estimation (KDE) etc.

This approach largely avoids the above mentioned perils of NHST, except for the third one on complex performance measures. However, it is known that the value of Bayes factor can be very sensitive to the choice of prior distribution in the alternative model [9]. Another problem with Bayes factor is that the null hypothesis can be strongly preferred even with very few data and very large uncertainty in the estimate of the performance difference [9]. Furthermore, generally speaking, a single Bayes factor is much less informative than the entire posterior probability distribution of the performance difference provided by our Bayesian estimation based approach.

2) Bayesian Estimation: It has been loudly advocated in recent years that the Bayesian estimation approach to comparing two groups of data has many advantages over using NHST or Bayes factor [8], [9]. However, to our knowledge, almost all the existing models of Bayesian

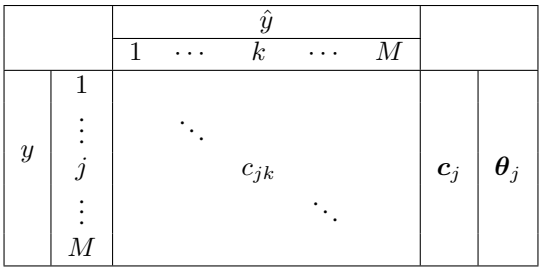


Figure 1: A schematic diagram of confusion matrix.

estimation deal with continuous values (that can be described by Gaussian or t distributions) but not discrete classification outcomes, and they produce estimations for simple statistics (such as the average difference between the two given groups) but not complex performance measures (such as the F_1 score). Probably the most closely related work is that of Goutte and Gaussier [21] (which has been brought to our attention by the reviewers). Their F_1 score model constructed using a couple of Gamma variates is not as expressive and flexible as ours. It is restricted to a single F_1 score for binary classification with two classes only. In contrast, our proposed approach opens up many possibilities for adaptation or extension.

III. Our Approach

A. Probabilistic Models

Let us consider a multi-class classifier which has been tested on a collection of N labelled test documents, $\mathcal{D}$. Here we focus on the setting of multi-class single-label (aka "one-of") classification where one document belongs to one and only one class [1], [2]. For each document $\boldsymbol{x}_i$ $(i = 1, \ldots, N)$, we have its true class label y_i as well as its predicted class label $\hat{y}_i$. Given that there are M different classes, the classification results could be fully summarised into an $M \times M$ confusion matrix $\boldsymbol{C}$ where the element c_{jk} at the j-th row and the k-th column represents the number of documents with true class label j but predicted class label k, as shown in Fig. 1.

The performance measures miF_1 and maF_1 can be calculated straightforwardly based on such a confusion matrix. However, as we have explained earlier, we are not satisfied with knowing only a single score value of the performance measure, but instead would like to treat the performance measure (either miF_1 or maF_1) as a random variable ψ and estimate its uncertainty by examining its posterior probability distribution.

The test documents can usually be considered as "independent trials", i.e., their true class labels y_i are independent and identically distributed (i.i.d.). For each test document, we use $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_M)$ to represent the probabilities that it truly belongs to each class: $\mu_j = \Pr[y_i = j]$ $(j = 1, \ldots, M)$, $\sum_{j=1}^{M} \mu_j = 1$. This means that the class sizes $\boldsymbol{n} = (n_1, \ldots, n_M)$ would follow a Multinomial distribution with parameter N and $\boldsymbol{\mu}$: $\boldsymbol{n} \sim \text{Mult}(N, \boldsymbol{\mu})$, i.e.,

$$\Pr[\boldsymbol{n}|N, \boldsymbol{\mu}] = \frac{N!}{n_1! \ldots n_M!} \prod_{j=1}^{M} \mu_j^{n_j} = \frac{\Gamma\left(\sum_{j=1}^{M}(n_j+1)\right)}{\prod_{j=1}^{M}\Gamma(n_j+1)} \prod_{j=1}^{M} \mu_j^{n_j} \ . \quad (1)$$

It would then be convenient to use the Dirichlet distribution (which is conjugate to the Multinomial distribution) as the prior distribution of parameter $\boldsymbol{\mu}$. More specifically, $\boldsymbol{\mu} \sim \text{Dir}(\boldsymbol{\beta})$, i.e.,

$$\Pr[\boldsymbol{\mu}] = \frac{\Gamma\left(\sum_{j=1}^{M}\beta_j\right)}{\prod_{j=1}^{M}\Gamma(\beta_j)} \prod_{j=1}^{M} \mu_j^{\beta_j - 1} \ , \quad (2)$$

where the hyper-parameter $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_M)$ encodes our prior belief about each class's proportion in the test document collection. If we do not have any prior knowledge, we can simply set $\boldsymbol{\beta} = (1, \ldots, 1)$ that yields a uniform distribution, as we did in our experiments.

Furthermore, let $\boldsymbol{c}_j = (c_{j1}, \ldots, c_{jM})$ denote the j-th row of the confusion matrix. In other words, $\boldsymbol{c}_j$ shows how those documents belonging to class j are classified. For each test document from that class j, we use $\boldsymbol{\theta}_j = (\theta_{j1}, \ldots, \theta_{jM})$ to represent the probabilities that it is classified into different classes: $\theta_{jk} = \Pr[\hat{y}_i = k | y_i = j]$ $(k = 1, \ldots, M)$, $\sum_{k=1}^{M} \theta_{jk} = 1$. This means that for each class j, the corresponding vector $\boldsymbol{c}_j$ would follow a Multinomial distribution with parameter n_j and $\boldsymbol{\theta}_j$: $\boldsymbol{c}_j \sim \text{Mult}(n_j, \boldsymbol{\theta}_j)$, i.e.,

$$\Pr[\boldsymbol{c}_j|n_j, \boldsymbol{\theta}_j] = \frac{n_j!}{c_{j1}! \ldots c_{jM}!} \prod_{k=1}^{M} \theta_{jk}^{c_{jk}} = \frac{\Gamma\left(\sum_{k=1}^{M}(c_{jk}+1)\right)}{\prod_{k=1}^{M}\Gamma(c_{jk}+1)} \prod_{k=1}^{M} \theta_{jk}^{c_{jk}} \ . \quad (3)$$

It would then be convenient to use the Dirichlet distribution (which is conjugate to the Multinomial distribution) as the prior distribution of parameter $\boldsymbol{\theta}_j$. More specifically, $\boldsymbol{\theta}_j \sim \text{Dir}(\boldsymbol{\omega}_j)$, i.e.,

$$\Pr[\boldsymbol{\theta}_j] = \frac{\Gamma\left(\sum_{k=1}^{M}\omega_{jk}\right)}{\prod_{k=1}^{M}\Gamma(\omega_{jk})} \prod_{k=1}^{M} \theta_{jk}^{\omega_{jk} - 1} \ , \quad (4)$$

where the hyper-parameter $\boldsymbol{\omega}_j = (\omega_{j1}, \ldots, \omega_{jM})$ encodes our prior belief about a classifier's prediction accuracy for class j.

Moreover, we introduce a higher-level overarching hyper-parameter η to capture the overall tendency of making correct predictions. In other words, we assume that ω_{jj}, the prior probability of correctly assigning a document of class j to its true class j, would be governed by the same

hyper-parameter η for all the classes. Thus for class j, we set $\omega_{jk} = \eta$ if $j = k$ and $\omega_{jk} = (1-\eta)/(M-1)$ if $j \neq k$. The hyper-parameter η itself can be considered as a random variable that ranges between 0 and 1, so it is assumed to follow a Beta distribution. More specifically, $\eta \sim \text{Beta}(\boldsymbol{\alpha})$, i.e.,

$$\Pr[\eta] = \frac{\Gamma(\alpha_1, \alpha_2)}{\Gamma(\alpha_1)\Gamma(\alpha_2)} \mu^{\alpha_1 - 1}(1-\mu)^{\alpha_2 - 1} \ , \tag{5}$$

where the hyper-parameter $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)$ encodes our prior belief about the average classification accuracy of the given classifier. If we do not have such knowledge, we can simply set $\boldsymbol{\alpha} = (1, 1)$ that yields a uniform distribution, as we did in our experiments. Such a Bayesian *hierarchical* model [7], [8] is more powerful as it allows us to "share statistical strength" across different classes.

Once the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\theta}_j$ $(j = 1, \ldots, M)$ have been estimated, it will be easy to calculate, for each class, the contingency table of "expected" prediction results: true positive (tp), false positive (fp), true negative (tn), and false negative (fn). For example, the anticipated number of true positive predictions, for class j, should be the number of test documents belonging to that class $N\mu_j$ times the rate of being predicted by the classifier into that class as well θ_{jj}. The equations to calculate the contingency table for each class j are listed as follows.

$$tp_j = N\mu_j\theta_{jj} \qquad fp_j = \sum_{u \neq j} N\mu_u\theta_{uj}$$
$$fn_j = \sum_{v \neq j} N\mu_j\theta_{jv} \qquad tn_j = \sum_{u \neq j}\sum_{v \neq j} N\mu_u\theta_{uv}$$

In *micro-averaging*, we pool the per-document predictions across classes, and then use the pooled contingency table to compute the micro-averaged precision P, micro-averaged recall R, and finally their harmonic mean miF_1 as follows.

$$P = \frac{\sum_{j=1}^{M} tp_j}{\sum_{j=1}^{M}(tp_j + fp_j)} = \sum_{j=1}^{M} \mu_j\theta_{jj} \tag{6}$$

$$R = \frac{\sum_{j=1}^{M} tp_j}{\sum_{j=1}^{M}(tp_j + fn_j)} = \sum_{j=1}^{M} \mu_j\theta_{jj} \tag{7}$$

$$\text{mi}F_1 = \frac{2PR}{P+R} = \sum_{j=1}^{M} \mu_j\theta_{jj} \tag{8}$$

It is a well-known fact that in multi-class single-label (aka "one-of") classification, mi$F_1 = P = R$ which is actually identical to the overall accuracy of classification [2].

In *macro-averaging*, we use the contingency table of each individual class j to compute that particular class's precision P_j as well as recall R_j, and finally compute a simple average of the F_1 scores over classes to get maF_1 as follows.

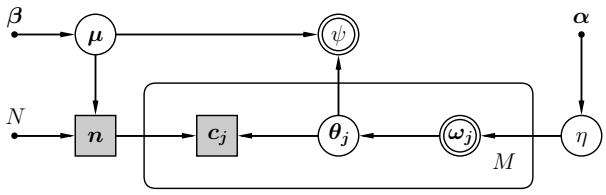


Figure 2: The probabilistic graphical model for estimating the uncertainty of average F_1 scores.

$$P_j = \frac{tp_j}{tp_j + fp_j} = \frac{\mu_j\theta_{jj}}{\sum_{u=1}^{M}\mu_u\theta_{uj}} \tag{9}$$

$$R_j = \frac{tp_j}{tp_j + fn_j} = \theta_{jj} \tag{10}$$

$$\text{ma}F_1 = \left(\sum_{j=1}^{M} \frac{2P_jR_j}{P_j + R_j}\right)/M \tag{11}$$

In the above calculation of miF_1 and maF_1, N has been cancelled out so it does not appear in the final formulae. Therefore the deterministic variable ψ for the performance measure of interest (either miF_1 or maF_1) is a function that depends on $\boldsymbol{\mu}$ and $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_M$ only:

$$\psi = f(\boldsymbol{\mu}, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_M) \ . \tag{12}$$

The above model describes the generative mechanism of a multi-class classifier's test results (i.e., confusion matrix). It is summarised as follows, and also depicted in Fig. 2 as a probabilistic graphical model (PGM) [22] using common notations.

$$\begin{aligned}
\boldsymbol{\mu} &\sim \text{Dir}(\boldsymbol{\beta}) \\
\boldsymbol{n} &\sim \text{Mult}(N, \boldsymbol{\mu}) \\
\eta &\sim \text{Beta}(\boldsymbol{\alpha}) \\
\boldsymbol{\omega}_{jk} &= \begin{cases} \eta & \text{if } k = j \\ (1-\eta)/(M-1) & \text{if } k \neq j \end{cases} \text{ for } k = 1, \ldots, M \\
&\quad \text{for } j = 1, \ldots, M \\
\boldsymbol{\theta}_j &\sim \text{Dir}(\boldsymbol{\omega}_j) \qquad \text{for } j = 1, \ldots, M \\
\boldsymbol{c}_j &\sim \text{Mult}(n_j, \boldsymbol{\theta}_j) \text{ for } j = 1, \ldots, M \\
\psi &= f(\boldsymbol{\mu}, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_M)
\end{aligned}$$

In order to make a comparison of average F_1 scores between two classifiers A and B, we just need to build an model as described above for each classifier's miF_1 or maF_1, and then introduce yet another deterministic variable δ to represent their difference:

$$\delta = \psi^{\text{A}} - \psi^{\text{B}} \ . \tag{13}$$

The posterior probability distribution of δ provides comprehensive information on all aspects of the performance difference between A and B.

When applying our models to performance comparison, we first need to define a Region of Practical Equivalence (ROPE) [8], [9] for the performance difference δ around its null value 0, e.g., $[-0.05, +0.05]$, which encloses those values of δ deemed to be negligibly different from its null value for practical purposes. The size of the ROPE should be determined based on the specifics of the application domain.

B. Decision Rules

Given the posterior probability distribution of δ, we can then reach a discrete judgement about how those two classifiers A and B compare with each other by examining the relationship between the 95% Highest Density Interval (HDI) of δ and the user-defined ROPE of δ [8], [9]. The 95% HDI is a useful summary of where the bulk of the most credible values of δ falls: by definition, every value inside the HDI has higher probability density than any value outside the HDI, and the total mass of points inside the 95% HDI is 95% of the distribution.

The decision rules are as follows:

- if the HDI sits fully within the ROPE, A is practically equivalent ($\approx$) to B;
- if the HDI sits fully at the left or right side of the ROPE, A is *significantly* worse ($\ll$) or better ($\gg$) than B respectively;
- if the HDI sits mainly though not fully at the left or right side of the ROPE, A is *slightly* worse ($<$) or better ($>$) than B respectively, but more experimental data would be needed to make a reliable judgement.

C. Software Implementation

The purpose of building these models for classification results is to assess the Bayesian posterior probability of δ — the performance difference between two classifiers A and B. An approximate estimation of δ can be obtained by sampling from its posterior probability distribution via Markov Chain Monte Carlo (MCMC) [7], [8] techniques.

We have implemented our models with an MCMC method *Metropolis-Hastings sampling* [7], [8]. The default configuration is to generate 50,000 samples, with no "burn-in", "lag", or "multiple-chains". It has been argued in the MCMC literature that those tricks are often unnecessary; it is perfectly correct to do a single long sampling run and keep all samples [22]–[24].

The program is written in Python utilising the module `PyMC3`[1] [25] for MCMC based Bayesian model fitting. The source code will be made open to the research community on the first author's homepage[2]. It is free, easy to use, and extensible to more sophisticated models (see Section V).

[1]http://pymc-devs.github.io/pymc3/

[2]http://www.dcs.bbk.ac.uk/~dell/

Table I: The filtered & vectorised `20newsgroups` dataset.

category	name	#train	#test
0	alt.atheism	480	319
1	comp.graphics	584	389
2	comp.os.ms-windows.misc	591	394
3	comp.sys.ibm.pc.hardware	590	392
4	comp.sys.mac.hardware	578	385
5	comp.windows.x	593	395
6	misc.forsale	585	390
7	rec.autos	594	396
8	rec.motorcycles	598	398
9	rec.sport.baseball	597	397
10	rec.sport.hockey	600	399
11	sci.crypt	595	396
12	sci.electronics	591	393
13	sci.med	594	396
14	sci.space	593	394
15	soc.religion.christian	599	398
16	talk.politics.guns	546	364
17	talk.politics.mideast	564	376
18	talk.politics.misc	465	310
19	talk.religion.misc	377	251

IV. Experiments

A. Dataset

We have conducted our experiments on a standard benchmark dataset for text classification, `20newsgroups`[3]. In order to ensure the reproducibility of our experimental results, we choose to use not the raw document collection, but a publicly-available ready-made "vectorised" version[4]. It has been split into training (60%) and testing (40%) subsets by date rather than randomly. Following the recommendation of the provider, this dataset has also been "filtered" by striping newsgroup-related metadata (including headers, footers, and quotes). Therefore our reported F_1 scores will look substantially lower than those in the related literature, but such an experimental setting is much more realistic and meaningful. Table I shows the number of training documents and the number of testing documents for each category in this dataset.

B. Classifiers

In the experiments, we have applied our proposed approach to carefully analyse the performances of two well-known supervised machine learning algorithms that are widely used for real-world text classification tasks: Naive Bayes (NB) and linear Support Vector Machine (SVM) [2]. For the former, we consider its two common variations: one with the Bernoulli event model (NB_{Bern}) and the other with the Multinomial event model (NB_{Mult}) [26], [27]. For the latter, we consider its two common variations: one with the $L1$ penalty (SVM_{L1}) and the other with the $L2$ penalty (SVM_{L2}) [28], [29]. Thus we have four different classifiers in total. Obviously, the classification results of

[3]http://qwone.com/~jason/20Newsgroups/

[4]http://scikit-learn.org/stable/datasets/twenty_newsgroups.html

Table II: The F_1 scores of the classifiers being compared.

category	NB_{Bern}	NB_{Mult}	SVM_{L1}	SVM_{L2}
0	0.398	0.480	0.453	0.480
1	0.551	0.666	0.626	0.638
2	0.170	0.577	0.599	0.605
3	0.545	0.641	0.591	0.611
4	0.431	0.682	0.654	0.676
5	0.666	0.768	0.709	0.717
6	0.645	0.781	0.716	0.768
7	0.634	0.725	0.561	0.695
8	0.596	0.741	0.720	0.735
9	0.770	0.850	0.760	0.634
10	0.840	0.731	0.834	0.846
11	0.632	0.725	0.737	0.742
12	0.533	0.631	0.523	0.557
13	0.681	0.796	0.716	0.736
14	0.647	0.756	0.694	0.706
15	0.655	0.682	0.656	0.691
16	0.535	0.641	0.569	0.600
17	0.694	0.797	0.759	0.743
18	0.398	0.487	0.457	0.472
19	0.207	0.248	0.319	0.311
$\text{mi}F_1$	0.581	0.689	0.641	0.660
$\text{ma}F_1$	0.561	0.670	0.633	0.648

NB_{Bern} and NB_{Mult} would be highly correlated, and those of SVM_{L1} and SVM_{L2} as well. Among them, SVM_{L2} is widely regarded as the state-of-the-art text classifier [1], [4], [30]. It is also worth to notice that the NB algorithms will be applied not to the raw bag-of-words text datasets as people usually do, but on the vectorised `20newsgroups` dataset which has already been transformed by TF-IDF term weighting and document length normalisation.

We have used the off-the-shelf implementation of these classification algorithms provided by a Python machine learning library `scikit-learn`[5] in our experiments, again for the reproducibility reasons. The smoothing parameter α for the NB algorithm and the regularisation parameter C for the linear SVM algorithm have been tuned via grid search with 5-fold cross-validation on the training data for the macro-averaged F_1 score. The optimal parameters found are: NB_{Bern} with $\alpha = 10^{-14}$, NB_{Mult} with $\alpha = 10^{-3}$, SVM_{L1} with $C = 2^2$, SVM_{L2} with $C = 2^1$.

C. Results

The F_1 scores of these four classifiers in comparison, for each target category as well as the micro-average (miF_1) and the macro-average (maF_1), are shown in Table II. Please be reminded that the `20newsgroups` dataset used in our experiments is more challenging than most of its versions appeared in literature.

The confusion matrices of the classifiers provide all the data that our model needs to make comparison of average F_1 scores. They are shown in Fig. 3 to ensure the reproducibility of our experimental results.

[5]http://scikit-learn.org/stable/

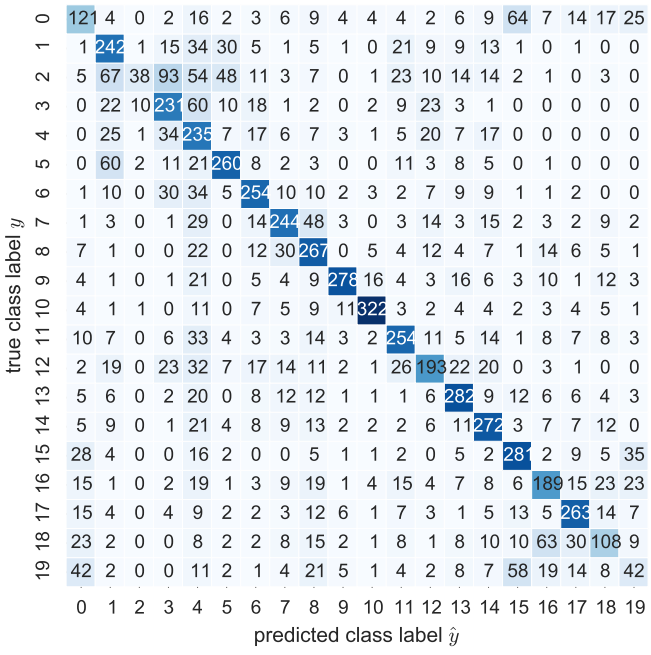
Table III: Frequentist comparison of the average F_1 scores.

		sign-test	t-test
NB_{Bern} vs NB_{Mult}	$\text{mi}F_1$	0.000	0.000
	$\text{ma}F_1$	0.000	0.000
SVM_{L1} vs SVM_{L2}	$\text{mi}F_1$	0.000	0.013
	$\text{ma}F_1$	0.003	0.145
NB_{Mult} vs SVM_{L2}	$\text{mi}F_1$	0.000	0.000
	$\text{ma}F_1$	0.115	0.138

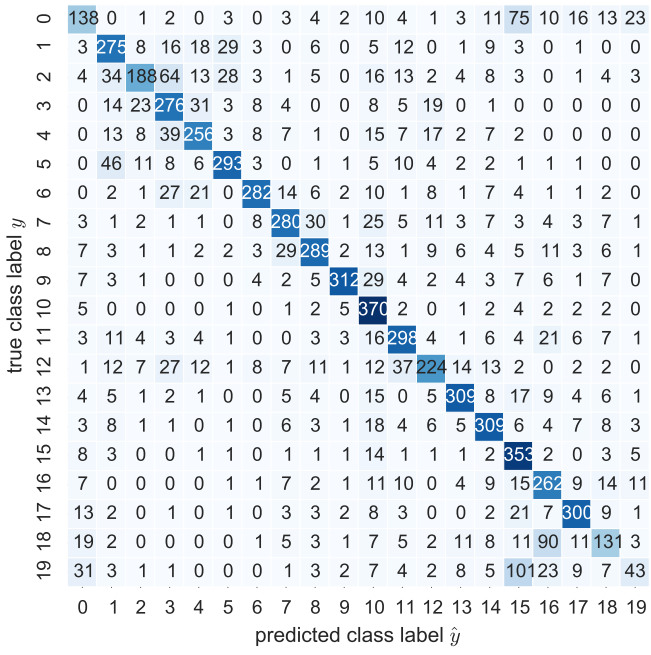
Table III shows the results of frequentist comparison of the average F_1 scores. The column "sign-test" contains the two-sided p-values of using sign-test on the micro level (called s-test in [4]) and also the macro level (called S-test in [4]). The column "t-test" contains the two-sided p-values of using unpaired t-test on the micro level (called p-test in [4]) and using paired t-test on the macro level (called T-test in [4]). The above tests on the macro level are based on the classes' F_1 scores, but those on the micro level are actually based on the accuracy of document-class assignments which happens to be equivalent to $\text{mi}F_1$ in multi-class single-label (aka "one-of") classification [2]. However, those tests will not longer be feasible on the micro level if we extend to multi-class multi-label (aka "any-of") classification.

Table IV shows the results of Bayesian comparison of the average F_1 scores, where the ROPE is set to $[-0.005, +0.005]$. It can be clearly seen that our proposed Bayesian approach offers a lot richer information about the difference between two classifiers' average F_1 scores than the frequentist approach does. In addition to the final judgement ("derision") of comparison, we have shown the posterior "mean", standard deviation ("std"), Monte Carlo error ("MC error"), the Bayes factor estimated by the SD method ("BF_{SD}"), the percentage lower or greater than the null value 0 ("LG pct"), the percentage covered by the ROPE ("ROPE pct"), and the 95% "HDI". By contrast, the frequentist NHST based approach would lead to a far less complete picture as it has only the p-values (and maybe also the confidence intervals) to offer. Furthermore, note that the judgements made by the Bayesian estimation on several cases are different from those made by the frequentist NHST (e.g., at the significance level 5%). So even if in some researchers' opinion the superiority of the former over the latter is still debatable, there is no doubt that the former can at least be complementary to the latter.

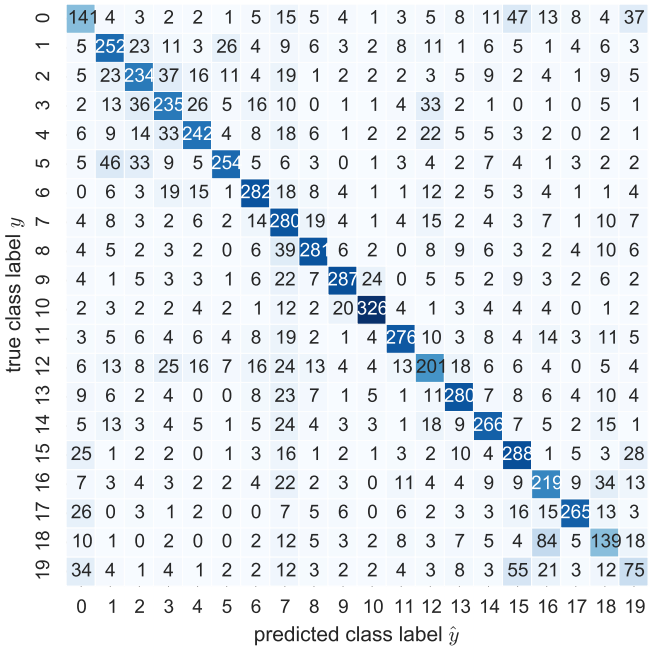
The Bayesian comparisons of $\text{ma}F1$ for three pairs of text classifiers (NB_{Bern} vs NB_{Mult}, SVM_{L1} vs SVM_{L2}, and NB_{Mult} vs SVM_{L2}) are visualised in Fig. 4, 5, and 6 respectively. In each figure, the "trace plot" sub-graph shows the corresponding MCMC trace which proves the convergence of sampling; the "posterior plot" sub-graph shows the posterior probability distribution of the performance difference variable; and the "factor plot" sub-graph



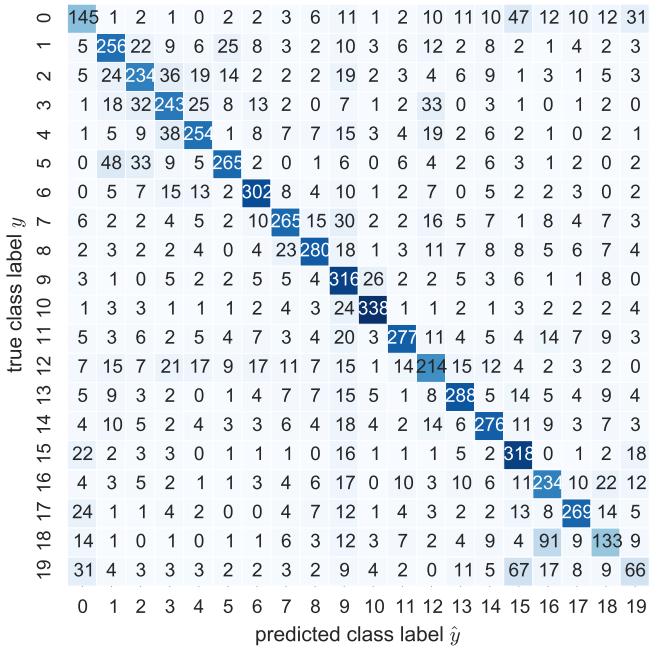
(a) NB$_{Bern}$



(b) NB$_{Mult}$



(c) SVM$_{L1}$



(d) SVM$_{L2}$

Figure 3: The confusion matrices of the classifiers to be compared in our experiments.

shows the estimation of the Bayes factor by the SD method (see Section II-B1). The figures for mi$F1$ look very similar to those for maF_1, so they are omitted due to the space limit.

The results of our Bayesian comparison between NB$_{Bern}$ and NB$_{Mult}$ indicate that NB$_{Bern}$ is significantly outperformed by NB$_{Mult}$ in terms of both miF_1 and maF_1. This confirms the finding of [26] on this harder dataset.

The results of our Bayesian comparison between SVM$_{L1}$ and SVM$_{L2}$ indicate that SVM$_{L1}$ is only slightly outperformed by SVM$_{L2}$ in terms of both miF_1 and maF_1 though the former may have its advantages sparsity. This is complementary to the findings reported in [28].

The results of our Bayesian comparison between NB$_{Mult}$ and SVM$_{L2}$ — the better performing classifiers from the NB

Table IV: Bayesian comparison of the average F_1 scores.

		mean	std	MC error	BF_{SD}	LG pct	ROPE pct	HDI	**decision**
NB_{Bern} vs NB_{Mult}	miF_1	-0.107	0.008	0.000	0.000	100.0%<0<0.0%	0.0%	$[-0.122, -0.092]$	$\ll$
	maF_1	-0.109	0.008	0.000	0.000	100.0%<0<0.0%	0.0%	$[-0.123, -0.094]$	$\ll$
SVM_{L1} vs SVM_{L2}	miF_1	-0.020	0.008	0.000	0.097	99.4%<0<0.6%	2.9%	$[-0.035, -0.005]$	$<$
	maF_1	-0.016	0.008	0.000	0.177	98.0%<0<2.0%	7.3%	$[-0.031, -0.001]$	$<$
NB_{Mult} vs SVM_{L2}	miF_1	$+0.028$	0.008	0.000	0.003	0.0%<0<100.0%	0.1%	$[+0.013, +0.043]$	$\gg$
	maF_1	$+0.022$	0.008	0.000	0.027	0.2%<0<99.8%	1.3%	$[+0.007, +0.037]$	$\gg$

and SVM camps — indicate that NB_{Mult} works a lot better than SVM_{L2} in terms of both miF_1 and maF_1, which is a bit surprising given that SVM_{L2} is widely regarded as the state-of-the-art classifier for text classification. This somewhat supports the claim of [31] that NB_{Mult}, if properly enhanced by TF-IDF term weighting and document length normalisation, can reach a similar or even better performance compared to SVM_{L2}.

V. Extensions

Our probabilistic model for comparing the average F_1 scores has been described in the multi-class single-label (aka "one-of") classification setting, but it is readily extensible to the multi-class multi-label (aka "any-of") classification setting [1], [2]. In that case, the Dirichlet/Multinomial distributions should simply be replaced by multiple Beta/Binomial distributions each of which corresponds to one specific target class, because a multi-class multi-label classifier is nothing more than a composition of independent binary classifiers.

To compare classifiers using a performance measure different from the average F_1 score (miF_1 or maF_1), we would only need to replace the function $f(\boldsymbol{\mu}, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_M)$ for computing δ, as long as that performance measure could be calculated based on the classification confusion matrix alone. For example, it would be straightforward to extend our probabilistic model to handle the more general F_β measure ($\beta \geq 0$) with $\beta \neq 1$ [2], [3].

In this paper we have focused on comparing text classifiers, but our Bayesian estimation based approach can actually be used to compare classifiers on any type of data, e.g., images. Only the confusion matrices are needed for our models. It does not really matter what kind of data are classified.

VI. Conclusions

The main contribution of this paper is a Bayesian estimation approach to assessing the uncertainty of average F_1 scores in the context of multi-class text classification.

By modelling the full posterior probability distribution of miF_1 or maF_1, we are able to make meaningful *interval estimation* (e.g., the 95% HDI) instead of simplistic *point estimation* of a text classifier's future performance on unseen data. The rich information provided by our model allows us to make much more comprehensive performance comparison between text classifiers than what the traditional frequentist NHST can possibly offer.

References

[1] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002.

[2] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[3] C. J. van Rijsbergen, *Information Retrieval*, 2nd ed. London, UK: Butterworths, 1979.

[4] Y. Yang and X. Liu, "A re-examination of text categorization methods," in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Berkeley, CA, USA, 1999, pp. 42–49.

[5] T. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[6] D. Zhang, J. Wang, and X. Zhao, "Estimating the uncertainty of average F1 scores," in *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR)*, Northampton, MA, USA, 2015, in press.

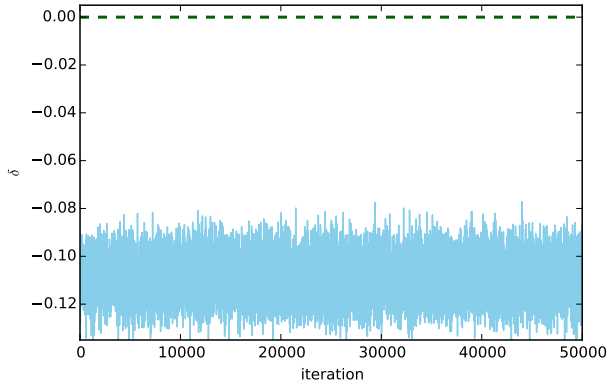
[7] A. Gelman, J. Carlin, H. Stern, D. Dunson, A. Vehtari, and D. Rubin., *Bayesian Data Analysis*, 3rd ed. Chapman & Hall/CRC, 2013.

[8] J. K. Kruschke, *Doing Bayesian Data Analysis: A Tutorial with R, JAGS, and Stan*, 2nd ed. Academic Press, 2014.

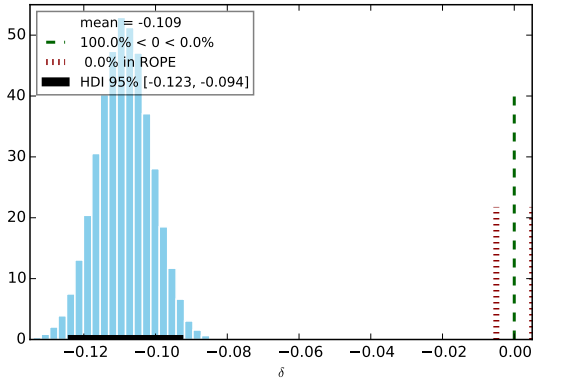
[9] ——, "Bayesian estimation supersedes the t test." *Journal of Experimental Psychology: General*, vol. 142, no. 2, p. 573, 2013.

[10] D. A. Hull, "Improving text retrieval for the routing problem using latent semantic indexing," in *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Dublin, Ireland, 1994, pp. 282–291.

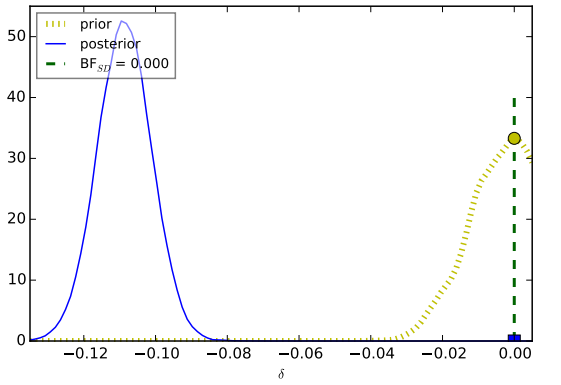
[11] H. Schutze, D. A. Hull, and J. O. Pedersen, "A comparison of classifiers and document representations for the routing problem," in *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, Seattle, WA, USA, 1995, pp. 229–237.



(a) trace plot

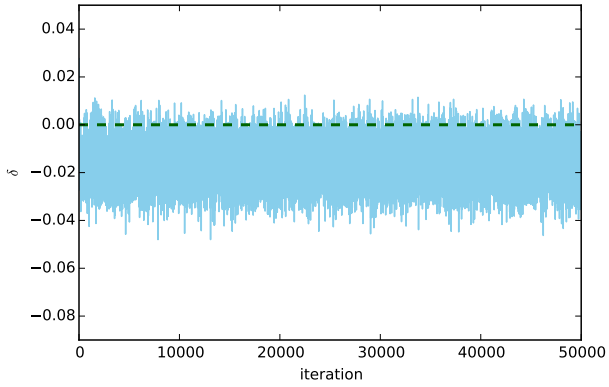


(b) posterior plot

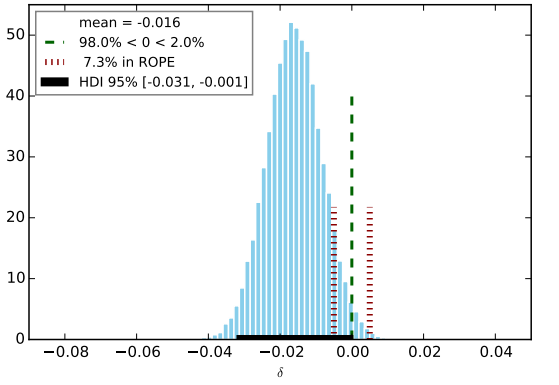


(c) factor plot

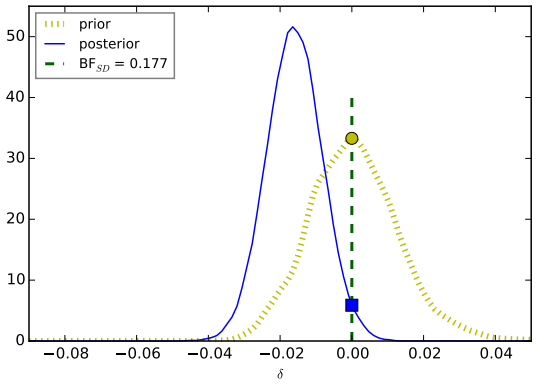
Figure 4: Comparing maF_1 between NB$_{Bern}$ and NB$_{Mult}$.



(a) trace plot



(b) posterior plot



(c) factor plot

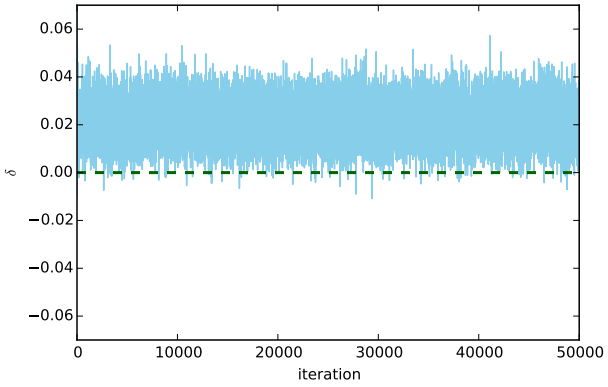
Figure 5: Comparing maF_1 between SVM$_{L1}$ and SVM$_{L2}$.

[12] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

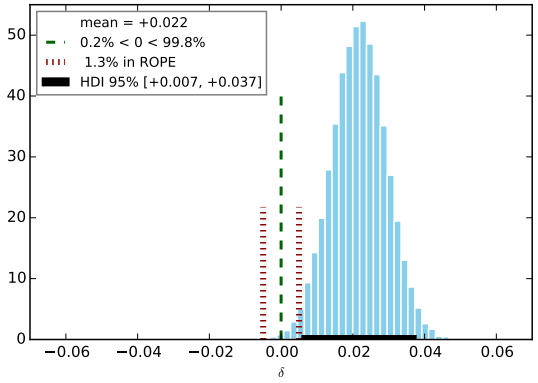
[13] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research (JMLR)*, vol. 7, pp. 1–30, 2006.

[14] D. Barber, "Are two classifiers performing equally? a treatment using Bayesian hypothesis testing," IDIAP, Tech. Rep., 2004.

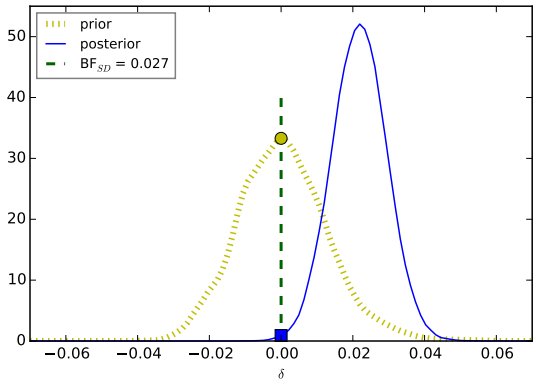
[15] ——, *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.



(a) trace plot



(b) posterior plot



(c) factor plot

Figure 6: Comparing maF_1 between NB$_{Mult}$ and SVM$_{L2}$.

[16] J. N. Rouder, P. L. Speckman, D. Sun, R. D. Morey, and G. Iverson, "Bayesian t tests for accepting and rejecting the null hypothesis," *Psychonomic Bulletin & Review*, vol. 16, no. 2, pp. 225–237, 2009.

[17] Z. Dienes, "Bayesian versus orthodox statistics: Which side are you on?" *Perspectives on Psychological Science*, vol. 6, no. 3, pp. 274–290, 2011.

[18] J. M. Dickey and B. P. Lientz, "The weighted likelihood ratio, sharp hypotheses about chances, the order of a markov chain," *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 214–226, 1970.

[19] R. Wetzels, J. G. Raaijmakers, E. Jakab, and E.-J. Wagenmakers, "How to quantify support for and against the null hypothesis: A flexible WinBUGS implementation of a default Bayesian t test," *Psychonomic Bulletin & Review*, vol. 16, no. 4, pp. 752–760, 2009.

[20] E.-J. Wagenmakers, T. Lodewyckx, H. Kuriyal, and R. Grasman, "Bayesian hypothesis testing for psychologists: A tutorial on the Savage-Dickey method," *Cognitive Psychology*, vol. 60, no. 3, pp. 158–189, 2010.

[21] C. Goutte and E. Gaussier, "A probabilistic interpretation of precision, recall and F-score, with implication for evaluation," in *Proceedings of the 27th European Conference on IR Research (ECIR)*, Santiago de Compostela, Spain, 2005, pp. 345–359.

[22] D. Koller and N. Friedman, *Probabilistic Graphical Models - Principles and Techniques*. MIT Press, 2009.

[23] P. Resnik and E. Hardisty, "Gibbs sampling for the uninitiated," Defense Technical Information Center (DTIC), Tech. Rep. LAMP-TR-153, 2010.

[24] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[25] A. Patil, D. Huard, and C. J. Fonnesbeck, "PyMC: Bayesian stochastic modelling in Python," *Journal of Statistical Software*, vol. 35, no. 4, pp. 1–81, 2010.

[26] A. McCallum and K. Nigam, "A comparison of event models for Naive Bayes text classification," in *AAAI-98 Workshop on Learning for Text Categorization*, Madison, WI, 1998, pp. 41–48.

[27] V. Metsis, I. Androutsopoulos, and G. Paliouras, "Spam filtering with Naive Bayes - which Naive Bayes?" in *Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, USA, 2006, pp. 27–28.

[28] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani, "1-norm support vector machines," in *Advances in Neural Information Processing Systems (NIPS) 16*, vol. 16, Vancouver and Whistler, Canada, 2003, pp. 49–56.

[29] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[30] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *Journal of Machine Learning Research (JMLR)*, vol. 5, pp. 361–397, 2004.

[31] J. D. Rennie, L. Shih, J. Teevan, and D. R. Karger, "Tackling the poor assumptions of Naive Bayes text classifiers," in *Proceedings of the 20th International Conference on Machine Learning (ICML)*, Washington, DC, USA, 2003, pp. 616–623.